МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №2**

по дисциплине: Теория информации

тема: «Исследование кодов Шеннона-Фано»

Выполнил: ст. группы ПВ-223

Дмитриев Андрей Александрович

Проверил:

Твердохлеб В.В.

Белгород 2024 г.

**Цель лабораторной работы:** изучить способ кодирования сообщений по методу Шеннона-Фано. Научиться составлять код Шеннона-Фано для данного сообщения. Узнать, как вычисляется коэффициент сжатия и величина дисперсии. Сравнить метод Хаффмана с методом Шеннона-Фано по показателям сжатия и дисперсии.

## Выполнение заданий:

**№1.**

Построить код для сообщения, содержащего строку панграммы «в чащах юга жил бы цитрус? да но фальшивый экземпляр!». Для полученного кода рассчитать показатели коэффициента сжатия и дисперсии.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Символ | Вероятность | Этапы | | | | | | Код |
| I | II | III | IV | V | VI |
| *‘ ‘* | 9/54 | 0 | 0 | 0 |  |  |  | 000 |
| а | 5/54 | 1 |  |  |  | 001 |
| и | 3/54 | 1 | 0 | 0 |  |  | 0100 |
| л | 3/54 | 1 |  |  | 0101 |
| в | 2/54 | 1 | 0 | 0 |  | 01100 |
| р | 2/54 | 1 |  | 01101 |
| ы | 2/54 | 1 | 0 |  | 01110 |
| б | 1/54 | 1 |  | 01111 |
| г | 1/54 | 1 | 0 | 0 | 0 | 0 |  | 10000 |
| д | 1/54 | 1 | 0 | 100010 |
| е | 1/54 | 1 | 100011 |
| ж | 1/54 | 1 | 0 | 0 | 100100 |
| з | 1/54 | 1 | 100101 |
| й | 1/54 | 1 | 0 | 100110 |
| к | 1/54 | 1 | 100111 |
| м | 1/54 | 1 | 0 | 0 |  | 10100 |
| н | 1/54 | 1 | 0 | 101010 |
| о | 1/54 | 1 | 101011 |
| п | 1/54 | 1 | 0 | 0 | 101100 |
| с | 1/54 | 1 | 101101 |
| т | 1/54 | 1 | 0 | 101110 |
| у | 1/54 | 1 | 101111 |
| ф | 1/54 |  |  | 0 | 0 |  | 11000 |
| х | 1/54 |  | 0 | 110011 |

1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ц | 1/54 |  | 1 | 0 |  |  | 1 | 110011 |
| ч | 1/54 | 1 | 0 |  | 11010 |
| ш | 1/54 | 1 | 0 | 11010 |
| щ | 1/54 | 1 | 110111 |
| ь | 1/54 | 1 | 0 | 0 |  | 11100 |
| э | 1/54 | 1 | 0 | 111010 |
| ю | 1/54 | 1 | 111011 |
| я | 1/54 | 1 | 0 | 0 | 111100 |
| , | 1/54 | 1 | 111101 |
| ? | 1/54 | 1 | 0 | 111110 |
| ! | 1/54 | 1 | 111111 |

n = 53 - количество символов в сообщении

Всего в алфавите 35 символов. По формуле 𝑁 = 2𝐼, 𝑁 - мощность алфавита,

𝐼 - кол-во бит, необходимо для кодирования символа алфавита с помощью двоичного кода.

𝐼 = log2 𝑁; 𝐼 = log2 35 ≈ 6

𝐵 = 𝑛 × 𝜂, где 𝑛 - количество символов в сообщении, 𝜂 - необходимый объем бит для представления одного символа, в данном случае 𝜂 = 𝐼 = 6, 𝑛

= 54

𝐵 = 53 × 6 = 324

*B’ = 5+3+5+3+5+3+6+3+6+5+3+3+6+4+3+6+4+6+5+6+6+5+3+6+3+3+6+5+3+6+3+4+6+5+4+5+5+6+3+6+6+6+6+6+5+4+6+5+6 = 251*

*B*

*K* =

*comp B*′

# = 318

251

# ≈ 1,26

*lcp*. = ∑ *pi* ⋅ *li* =

*4,76*

*δ* = ∑ *pi* ⋅ (*li* − *lcp*.)2 = *1,48*

## №2.

Построить код для сообщения, содержащего строку «Victoria nulla est, Quam quae confessos animo quoque subjugat hostes» Для полученного кода рассчитать показатели коэффициента сжатия и дисперсии.

Victoria nulla est, Quam quae confessos animo quoque subjugat hostes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Символ | Вероятность | Этапы | | | | | | Код |
| I | II | III | IV | V | VI |
| ‘ ‘ | 9/68 | 0 | 0 | 0 |  |  |  | 000 |
| s | 7/68 | 1 |  |  |  | 001 |
| u | 7/68 | 1 | 0 |  |  |  | 010 |
| a | 6/68 | 1 | 0 |  |  | 0110 |
| o | 6/68 | 1 |  |  | 0111 |
| e | 5/68 | 1 | 0 | 0 | 0 |  |  | 1000 |
| q | 4/68 | 1 |  |  | 1001 |
| t | 4/68 | 1 | 0 |  |  | 1010 |
| i | 3/68 | 1 |  |  | 1011 |
| n | 3/68 | 1 | 0 | 0 | 0 |  | 11000 |
| c | 2/68 | 1 |  | 11001 |
| l | 2/68 | 1 | 0 |  | 11010 |
| m | 2/68 | 1 |  | 11011 |
| b | 1/68 | 1 | 0 | 0 | 0 | 111000 |
| f | 1/68 | 1 | 111001 |
| g | 1/68 | 1 | 0 | 111010 |
| h | 1/68 | 1 | 111011 |
| j | 1/68 | 1 | 0 | 0 | 111100 |
| r | 1/68 | 1 | 111101 |
| v | 1/68 | 1 | 0 | 111110 |
| , | 1/68 | 1 | 111111 |

n = 68 - количество символов в сообщении

Всего в алфавите 21 символ. По формуле 𝑁 = 2𝐼, 𝑁 - мощность алфавита, 𝐼

* кол-во бит, необходимо для кодирования символа алфавита с помощью двоичного кода.

𝐼 = log2 𝑁; 𝐼 = log2 21 ≈ 5

*B* = 𝑛 × 𝜂 , где 𝑛 - количество символов в сообщении, 𝜂 - необходимый объем бит для представления одного символа, в данном случае 𝜂 = 𝐼 = 5, 𝑛

= 68

*B* = 68 × 5 = 340 *δ* = ∑ *pi* ⋅ (*li* − *lcp*.)2 = 0,99

*B’ = 6+5+5+4+4+6+5+4+3+5+3+5+5+4+3+4+3+4+6+3+6+3+4+5+3+6 +3+4+4+3+5+4+5+6+4+3+3+4+3+3+4+3+4+3+4+4+4+4+3+3+3+6+4+6+4+4+3+5+4+3+4+4+3 = 278*

*B*

*K* =

*comp B*′

# = 340

278

# ≈ 1,22

*lcp*. = ∑ *pi* ⋅ *li* =

4,03



## №3.

Построить консольное приложение, реализующее процесс кодирования по методу Шеннона-Фано (с возможностью расчета коэффициента сжатия и дисперсии).

Листинг программы:

N = 6  
  
  
class Node:  
 def \_\_init\_\_(self, freqs):  
 self.freqs = freqs  
 self.left = None  
 self.right = None  
  
  
*# рассчет вероятностей символов*def calculate\_freq(s):  
 *# подсчет кол-ва каждого символа в строке* freqs = {}  
 for w in s:  
 freqs[w] = 0  
  
 for w in s:  
 freqs[w] += 1  
  
 *# расчет вероятностей* for symb in freqs:  
 freqs[symb] = round(freqs[symb] / len(s), 3)  
  
 freqs = sorted(freqs.items(), key=lambda item: item[1], reverse=True)  
  
 return freqs  
  
  
*# подсчет сумм вероятностей переданного списка*def calculate\_freqs\_sum(d):  
 c = 0  
  
 dd = dict(d)  
 for v in dd.values():  
 c += v  
  
 return c  
  
  
def printCodes(shannon\_codes):  
 print("\nПолученные коды:")  
  
 for key, value in shannon\_codes.items():  
 print(f"{key} : {value}")  
  
  
def printFreqs(freqs):  
 print("Таблица частот(вероятностей):")  
 for el in freqs:  
 print(f"{el[0]} : {el[1]}")  
  
  
def encode(input\_s, shannon\_codes):  
 encoded\_s = ""  
  
 for symb in input\_s:  
 encoded\_s += shannon\_codes[symb]  
  
 return encoded\_s  
  
  
*# расчет коэффициента сжатия*def calculateKcomp(input\_s, encoded\_s):  
 return len(input\_s) \* N / len(encoded\_s)  
  
  
*# расчет дисперсии*def calculateDispersion(freqs, shannonCodes):  
 dispersion = 0  
  
 *# рассчитаем Lср* l\_middle = 0  
 for f in freqs:  
 l\_middle += f[1] \* len(shannonCodes[f[0]])  
  
 for f in freqs:  
 dispersion += f[1] \* ((len(shannonCodes[f[0]]) - l\_middle) \*\* 2)  
  
 return dispersion  
  
  
*# проходом по дереву Шеннона-Фано получаем коды*def get\_codes(node, s, shannon\_codes):  
 if node.right is None:  
 shannon\_codes[node.freqs[0][0]] = s  
 return  
  
 get\_codes(node.left, s + "0", shannon\_codes)  
 get\_codes(node.right, s + "1", shannon\_codes)  
  
  
*# генерация дерева Шеннона-Фано*def createShannonTree(node):  
 if len(node.freqs) == 1:  
 return  
  
 freqs\_sum = 0  
 mid\_freqs\_sum = calculate\_freqs\_sum(node.freqs) / 2  
  
 *# поиск разделительного элемента в списке* sep\_index = -1  
 for i in range(len(node.freqs)):  
 freqs\_sum += node.freqs[i][1]  
  
 if freqs\_sum >= mid\_freqs\_sum:  
 if (freqs\_sum - mid\_freqs\_sum) <= (mid\_freqs\_sum - (freqs\_sum - node.freqs[i][1])):  
 *# freqs[i] - разделитель* sep\_index = i  
  
 else:  
 *# freqs[i-1] - разделитель* sep\_index = i - 1  
  
 break  
  
 node.right = Node(node.freqs[:sep\_index + 1])  
 node.left = Node(node.freqs[sep\_index + 1:])  
  
 createShannonTree(node.right)  
 createShannonTree(node.left)  
  
  
def main():  
 input\_s = input("Введите текст: ")  
  
 print(f"\nВведенный текст: (длина = {len(input\_s)})\n{input\_s}\n")  
  
 freqs = calculate\_freq(input\_s)  
  
 root = Node(freqs)  
  
 printFreqs(freqs)  
  
 createShannonTree(root)  
  
 shannon\_codes = {}  
  
 get\_codes(root, "", shannon\_codes)  
  
 printCodes(shannon\_codes)  
  
 encoded\_s = encode(input\_s, shannon\_codes)  
  
 print(f"\nЗакодированное сообщение: (длина = {len(encoded\_s)})\n{encoded\_s}")  
  
 print(f"\nКоэффициент сжатия: {calculateKcomp(input\_s, encoded\_s)}")  
 print(f"\nДисперсия: {calculateDispersion(freqs, shannon\_codes)}")  
  
  
main()

## Результат для первого сообщения:

C:\Users\dmitr\AppData\Local\Programs\Python\Python39\python.exe T:/2kurs2sem/InformTheor/lab2/main.py

Введите текст: в чащах юга жил бы цитрус? да но фальшивый экземпляр!

Введенный текст: (длина = 53)

в чащах юга жил бы цитрус? да но фальшивый экземпляр!

Таблица частот(вероятностей):

: 0.17

а : 0.094

и : 0.057

л : 0.057

в : 0.038

ы : 0.038

р : 0.038

ч : 0.019

щ : 0.019

х : 0.019

ю : 0.019

г : 0.019

ж : 0.019

б : 0.019

ц : 0.019

т : 0.019

у : 0.019

с : 0.019

? : 0.019

д : 0.019

н : 0.019

о : 0.019

ф : 0.019

ь : 0.019

ш : 0.019

й : 0.019

э : 0.019

к : 0.019

з : 0.019

е : 0.019

м : 0.019

п : 0.019

я : 0.019

! : 0.019

Полученные коды:

! : 000000

я : 000001

п : 00001

м : 000100

е : 000101

з : 000110

к : 000111

э : 001000

й : 001001

ш : 00101

ь : 001100

ф : 001101

о : 00111

н : 010000

д : 010001

? : 01001

с : 010100

у : 010101

т : 010110

ц : 010111

б : 011000

ж : 011001

г : 01101

ю : 011100

х : 011101

щ : 01111

ч : 10000

р : 10001

ы : 10010

в : 10011

л : 1010

и : 1011

а : 110

: 111

Закодированное сообщение: (длина = 251)

10011111100001100111111001110111101110001101110111011001101110101110110001001011101011110110101101000101010101010001001111010001110111010000001111110011011101010001100001011011100111001000100111100100000011100011000010100010000001101000000110001000000

Коэффициент сжатия: 1.2669322709163346

Дисперсия: 1.4822618449999996

Process finished with exit code 0

Результат для второго сообщения:

C:\Users\dmitr\AppData\Local\Programs\Python\Python39\python.exe T:/2kurs2sem/InformTheor/lab2/main.py

Введите текст: Victoria nulla est, Quam quae confessos animo quoque subjugat hostes

Введенный текст: (длина = 68)

Victoria nulla est, Quam quae confessos animo quoque subjugat hostes

Таблица частот(вероятностей):

: 0.132

u : 0.103

s : 0.103

o : 0.088

a : 0.088

e : 0.074

t : 0.059

i : 0.044

n : 0.044

q : 0.044

c : 0.029

l : 0.029

m : 0.029

V : 0.015

r : 0.015

, : 0.015

Q : 0.015

f : 0.015

b : 0.015

j : 0.015

g : 0.015

h : 0.015

Полученные коды:

: 111

u : 110

s : 101

o : 1001

a : 1000

e : 0111

t : 0110

i : 01011

n : 01010

q : 0100

c : 00111

l : 00110

m : 00101

V : 001001

r : 001000

, : 000111

Q : 000110

f : 000101

b : 000100

j : 000011

g : 000010

h : 00000

Закодированное сообщение: (длина = 278)

00100101011001110110100100100001011100011101010110001100011010001110111101011000011111100011011010000010111101001101000011111100111100101010000101011110110110011011111000010100101100101100111101001101001010011001111111011100001000000111100000101000011011100000100110101100111101

Коэффициент сжатия: 1.223021582733813

Дисперсия: 0.9995612159999998

Process finished with exit code 0

**№4.**

Получить кодовые представления сообщений из пунктов 1 и 2 задания по методу Хаффмана. Сравнить полученные результаты с методом Шеннона-Фано по показателям сжатия и дисперсии. Сделать соответствующие выводы.

По методу Хаффмана получим.

Для первого сообщения:

Коэффициент сжатия: 1.2669322709163346

Дисперсия: 2.118903524385903

Для второго сообщения:

Коэффициент сжатия: 1.4676258992805755

Дисперсия: 1.2569204152249132

Заметим, что коэффициент сжатия информации больше у Хаффмана, но дисперсия меньше.

**Вывод:** в ходе работы изучен способ кодирования сообщений по методу Шеннона-Фано. Получены навыки составления кода Шеннона-Фано для данных сообщений, нахождения коэффициента сжатия, величины дисперсии. Также мы сравнили метод Хаффмана с методом Шеннона- Фано по показателям сжатия и дисперсии.